

On the hardness of unlabeled multi-robot motion planning

Kiril Solovey* and Dan Halperin*
Blavatnic School of Computer Science
Tel Aviv University, Israel
email: {kirisol,danha}@post.tau.ac.il

Abstract—In *unlabeled* multi-robot motion planning several interchangeable robots operate in a common workspace. The goal is to move the robots to a set of target positions such that each position will be occupied by *some* robot. In this paper, we study this problem for the specific case of unit-square robots moving amidst polygonal obstacles and show that it is PSPACE-hard. We also consider three additional variants of this problem and show that they are all PSPACE-hard as well. To the best of our knowledge, this is the first hardness proof for the unlabeled case. Furthermore, our proofs can be used to show that the *labeled* variant (where each robot is assigned with a specific target position), again, for unit-square robots, is PSPACE-hard as well, which sets another precedence, as previous hardness results require the robots to be of different shapes.

I. INTRODUCTION

In practical settings where multiple robots operate in a common environment it is often the case that the robots are identical in form and functionality and thus are interchangeable. Specifically, in *unlabeled* multi-robot motion planning (unlabeled planning, in short) a group of identical robots need to reach a set of target positions. As the robots are identical we only require that in the end of the process each target position will be occupied by *some* robot. This is in contrast to the standard *labeled* (also known as *fully-colored*) multi-robot motion problem, where each robot is required to reach a *specific* target position, and the robots may differ in shape. While labeled planning has been of interest to many researchers for the past four decades, unlabeled planning has only been recently introduced and investigated.

A. Related work

We start with the much more intensively studied *labeled* case of multi-robot motion planning. Schwartz and Sharir [15] were the first to consider the labeled problem from the geometric point-of-view, and in particular studied the case of two discs moving amidst polygonal obstacles and developed an algorithm with a running time of $O(n^3)$, where n is the complexity of the workspace. Yap [24] also considered this setting and described an algorithm of complexity $O(n^2)$. Later on, Sharir and Sifrony [16] proposed an $O(n^2)$ algorithm as

well, although their algorithm deals with several additional types of robots, besides discs.

When the number of robots is no longer a fixed constant the problem can become computationally intractable. Specifically, Hopcroft et al. [8] showed that the problem is PSPACE-hard for the setting of rectangular robots bound to translate in a rectangular workspace. Their proof required the rectangular robots to be of varying dimensions. Spirakis and Yap [20] showed that the problem is NP-hard for disc robots in a simple-polygon workspace; here the proof strongly relies on the fact that the discs are of varying radii.

More recently, Hearn and Demaine [4, 5] improved the result of Hopcroft et al. by showing that the robots can be restricted to only two types— 2×1 and 1×2 rectangles. Their work is more general: They introduced in this work the *nondeterministic constraint logic* (NCL) model of computation, for which they describe several PSPACE-hard problems, and from which they derive the PSPACE-hardness of a variety of puzzle-like problems that consist of sliding game pieces. (We describe the NCL model in detail later on.) In particular, they applied their technique to the *SOKOBAN* puzzle, where multiple “crates” need to be pushed to target locations, and the *Rush Hour* game, where a parking attendant has to evacuate a car from a parking lot, by clearing a route blocked by other cars. Other hardness results using the NCL model have followed [2, 3, 7, 9].

Due to the various hardness results related to the problem, the interest of the Computational-Geometry community in the multi-robot motion planning problem has diminished over the years and the attention has gradually shifted to the Robotics community who started to develop sampling-based tools for the problem. Sampling-based algorithms¹ try to capture the structure of the configuration space of the problem (or more accurately, its division into “free” and “forbidden” regions) by random sampling of the space and connecting nearby configurations by “simple” paths, to form a *roadmap*. While such methods are incapable of determining whether a solution does not exist, they often provide asymptotic guarantees of completeness and optimality. Even though such techniques can be applied as-is to the labeled planning problem [14], many approaches that were specifically designed for the problem have emerged [6, 13, 18, 22, 23].

*This work has been supported in part by the Israel Science Foundation (grant no. 1102/11), by the German-Israeli Foundation (grant no. 1150-82.6/2011), and by the Hermann Minkowski-Minerva Center for Geometry at Tel Aviv University.

¹We only briefly mention this area of research, as it is beyond the scope of this paper.

Unlabeled multi-robot motion planning was introduced by Kloder and Hutchinson [10], who described a sampling-based algorithm for the problem. More recently, Solovey and Halperin have developed a sampling-based algorithm for the unlabeled problem, as well as for a generalization termed *k-color* planning that consists of k groups of interchangeable robots [17]. Krontiris et al. [11] describe an adaptation of this work for the problem of rearranging several objects using a robotic manipulator. Turpin et al. [21] describe an efficient and complete algorithm for unlabeled planning for disc robots which also guarantees finding the optimal solution in terms of the length of the longest path traversed by a robot. However, their algorithm makes the assumption that a certain portion of the free space, surrounding each start or target position, is *star-shaped*. More recently, Adler et al. [1] studied a similar setting with unlabeled disc robots operating in a simple polygon, although under a milder assumption requiring each pair of start or target positions to be separated by a distance of at least 4, where the radius of the robots is 1. They describe an algorithm with a running time of $O(n \log n + mn + m^2)$, where n is the complexity of the polygon and m is the number of robots. A crucial question that follows from their work is whether the efficient solution of the problem is possible due to the separation constraints or the fact that the robots are unlabeled.

B. Contribution

In this paper we study the problem of unlabeled multi-robot motion planning for unit-square robots moving amidst polygonal obstacles. We show that the decision problem, namely, to decide whether a solution exists, is PSPACE-hard. To the best of our knowledge, this is the first hardness proof for the unlabeled case. In fact we consider four variants of the unlabeled problem (see Section II for a precise definition) and show that they are all PSPACE-hard. For instance, we show that the seemingly easier version of the problem where only one of the robots is required to reach a certain target position while the other robots function as movable obstacles, is also computationally intractable. We mention that our proofs can be used to show that the labeled variant, again, for unit-square robots, is PSPACE-hard as well, which sets another precedence, as previous hardness results require the robots to be of different shapes [4, 8, 20]. The various hardness results for multi-robot motion planning are summarized in Table I. While our result in itself is negative, we hope that it will motivate research of other variants of the unlabeled problem which may turn out to be polynomially solvable.

This paper complements another work by the authors [19] in which we study a slightly different setting of the unlabeled problem and present an efficient algorithm to tackle it. In particular we consider the problem of unlabeled motion planning of unit-disc robots moving amid polygonal obstacles. We show that if two simplifying assumptions are made regarding the distances between pairs of start and target positions and between such positions and the obstacles, an efficient algorithm can be developed. In particular, our algorithm has a running

time² $\tilde{O}(m^4 + m^2 n^2)$, where m is the number of robots and n is the complexity of the obstacles. Furthermore, the algorithm returns a solution whose total length (namely the total length traveled by all the robots) is $\text{OPT} + 4m$, where OPT is the optimal solution cost. In spite of the difference in robots, we believe that the hardness result presented in the current paper hints that such mitigating separation (or other) assumption are essential in order to obtain efficient algorithms as in the other paper.

The organization of this paper is as follows. In Section II we describe the four variants of the unlabeled problem that will be considered in this paper. In Section III the NCL model of computation [4], which is a key ingredient in our hardness proof, is described. In Section IV we provide the hardness proofs.

II. PRELIMINARIES

Let r be a robot operating in the planar workspace \mathcal{W} . We denote by $\mathcal{C}(r)$ the *configuration space* of r , and by $\mathcal{F}(r) \subset \mathcal{C}(r)$ the *free space* of r —the collection of all configurations for which the robot does not collide with obstacles. Given $s, t \in \mathcal{F}(r)$, a *path* for r from s to t is a continuous function $\pi : [0, 1] \rightarrow \mathcal{F}(r)$, such that $\pi(0) = s, \pi(1) = t$.

We say that two robots r, r' are *geometrically identical* if $\mathcal{F}(r) = \mathcal{F}(r')$ for the same workspace \mathcal{W} . Let $R = \{r_1, \dots, r_m\}$ be a set of m geometrically identical robots, operating in a workspace \mathcal{W} . We use \mathcal{F} to denote $\mathcal{F}(r_i)$ for any $1 \leq i \leq m$.

Definition 1. A collection $C = \{c_1, \dots, c_m\}$ of m configurations is termed a *multi-configuration*. Such a multi-configuration is *free* if $C \subset \mathcal{F}$ and for every two distinct configurations $c, c' \in C$ it holds that $r(c) \cap r'(c') = \emptyset$, for $r, r' \in R$, where $r(x)$, for $x \in C$ denotes the area covered by some robot $r \in R$ when placed in x .

Definition 2. Given two free multi-configurations, $C = \{c_1, \dots, c_m\}, C' = \{c'_1, \dots, c'_m\}$, we say that they are *equivalent* ($C \equiv C'$) if there exist m paths $\Pi = \{\pi_1, \dots, \pi_m\}$ that move the m robots from C to C' . More formally, we demand that for every $c \in C$, there exists $1 \leq i \leq m$ for which $\pi_i(0) = c$; for every $c' \in C'$ there exists $1 \leq j \leq m$ for which $\pi_j(1) = c'$; for every $\tau \in [0, 1]$ the multi-configuration $\Pi(\tau) = \{\pi_1(\tau), \dots, \pi_m(\tau)\}$ is free.

Given two equivalent multi-configurations S, T we denote by $\Pi(S, T) = \{\pi_1, \dots, \pi_m\}$ a collection of m paths that move the robots from S to T by following the restrictions above. Note that Definition 2 requires that every target position will be occupied by *some* robot, in contrast with the classic definition which indicates which robot should reach where.

We define four decision problems that will each be shown to be PSPACE-hard below:

- 1) Given two free multi-configurations S, T , is it true that $S \equiv T$?

²For simplicity of presentation, log factors in the complexity of the algorithm are omitted, and hence the \tilde{O} notation is used.

Contributor	Problem	Complexity	Robots	Workspace
Hopcroft et al. [8]	labeled	PSPACE-hard	rectangles	rectangle
Spirakis, Yap [20]	labeled	strongly NP-hard	discs	simple polygon
Hearn, Demaine [4]	labeled	PSPACE-complete	1×2 rectangles	rectangle
this paper	unlabeled, labeled	PSPACE-hard	unit squares	polygonal obstacles

TABLE I: Hardness results related to the multi-robot problem.

- 2) Given a free multi-configuration S , and a configuration $t \in \mathcal{F}$, is there a multi-configuration T such that $t \in T$ and $S \equiv T$?
- 3) Given a free multi-configuration S , a configuration $s \in S$, and another configuration $t \in \mathcal{F}$, is there a multi-configuration T such that $t \in T$, $S \equiv T$ for which there exists $\pi \in \Pi(S, T)$ such that $\pi(0) = s, \pi(1) = t$?
- 4) Given two configurations $s, t \in \mathcal{F}$, are there two multi-configurations S, T such that $s \in S, t \in T$ and $S \equiv T$?

We will refer to these problems from now on as the *multi-to-multi*, *multi-to-single*, *multi-to-single-restricted*, and *single-to-single* problems, respectively. Note that *multi-to-single* differs from *multi-to-single-restricted* by allowing any robot to reach t . Although the four problems seem to be closely related, it is not clear whether it is possible to reduce one problem to another.

III. NONDETERMINISTIC CONSTRAINT LOGIC

In this section we review the *nondeterministic constraint logic* (NCL) model of computation, and state three decision problems that are based on this model and are shown to be PSPACE-complete [4]. An NCL machine is defined by a *constraint graph* $G = (V, E)$, a *weight function* $w : E \rightarrow \mathbb{N}$, and a *minimum-flow constraint* $c : V \rightarrow \mathbb{N}$.

Definition 3. A machine configuration is an orientation $o : E \rightarrow \{0, 1\}$ such that for every edge $(v, v') \in E$ it holds that $o(v, v') = 1, o(v', v) = 0$, or $o(v, v') = 0, o(v', v) = 1$. An orientation o is valid if for every $v \in V$ the sum of weights of the edges that are directed into v is at least the minimum-flow constraint of the vertex. More formally,

$$\forall v \in V : \sum_{v' \in N(v)} o(v', v) \cdot w(v', v) \geq c(v),$$

where $N(v)$ denotes the set of neighbors of v in G .

A *move* from one orientation to another consists of a reversal of the orientation of a single edge, while maintaining the minimum-flow constraints. Given two orientations o, o' we say that they are *equivalent*, and denote this relation by $o \equiv o'$, if o can be transformed into o' by a series of moves. Given these definition, the following decision problems are defined in [4]:

- 1) Given two orientations o_S, o_T , is it true that $o_S \equiv o_T$?
- 2) Given an orientation o_S and an edge $(v, v') \in E$ is there another orientation o_T such that $o_S \equiv o_T$ and $o_S(\{v, v'\}) \neq o_T(\{v, v'\})$, i.e., the orientation of (v, v') is reversed between the two configurations?
- 3) Let $(v, v'), (u, u') \in E$. Additionally, let $o_{(v, v')}, o_{(u, u')}$ be two orientations for these specific edges. Are there

two configurations o_S, o_T such that $o_S \equiv o_T$ and $o_S(v, v') = o_{(v, v')}, o_T(u, u') = o_{(u, u')}$?

These problems are termed *full-to-full*, *full-to-edge*, and *edge-to-edge*, respectively. We are interested in a particular setting of the problem where the constraint graph is planar and consists of only two types of vertices:

- An AND vertex has a minimum-flow constraint of two, and has exactly three incident edges, where one of the edges has a weight of two, and each of the other two edges has a weight of one.
- An OR vertex has a minimum-flow constraint of two and has exactly three incident edges, each with a weight of two.

The following Theorem will play a central role in our analysis. Its proof is found in [4, Theorem 12].

Theorem 4 (Hearn and Demaine). *Full-to-edge, full-to-full, and edge-to-edge, are PSPACE-complete, even when the constraints graph is simple, planar, and consists of only AND and OR vertices.*

A. Grid-embedded constraint graph

In order to simplify the reduction process in the following sections, we show that given a constraint graph G , as described above, it can be transformed into a two-dimensional constraint grid graph H , such that the three NCL decision problems remain PSPACE-hard on H as well. We mention that the authors of [4] use a similar grid embedding, but omit the relevant details. Thus we chose to provide a full description of this process here.

We generate a new constraint graph H whose vertices are grid vertices and edges are grid edges. Each edge of G is transformed into a noncrossing path in H . Such an embedding is possible due to the fact that G is simple and planar. For the purpose of the reduction it suffices to know that such an embedding can be carried out in polynomial time, but we mention that a linear-time algorithm by Liu et al. exists [12].

As G is planar and has a degree of three (it is exclusively made of AND and OR vertices), it can be embedded on a planar grid $H = (V_H, E_H)$ which is defined as follows. The set of vertices of H is defined to be $V_H := V \cup U$, where U is an additional set of vertices called *connectors* and where each $v \in V_H$ corresponds to a point on the grid. Every edge $(v, v') \in E_H$ corresponds to an axis-parallel segment that connects the two points v, v' on the grid. Given two vertices $v, v' \in V$ for which $(v, v') \in E$ we denote by $H(v, v') = (v, u_1, \dots, u_\ell, v')$ the path in H that corresponds to (v, v') .

We also define the weight and capacity functions w_H, c_H , respectively. Each vertex $v \in V$ maintains its original capacity

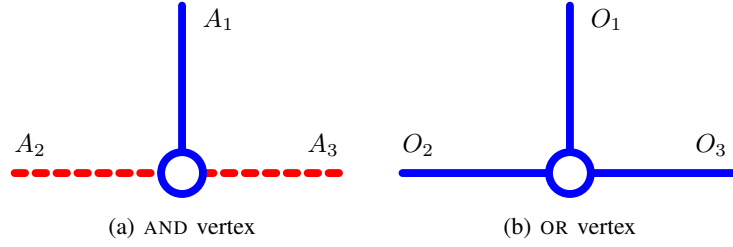


Fig. 1: AND and OR vertices. Red (dashed) edges have a weight of 1 and blue (solid) edges have a weight of 2. The blue (solid) vertex (circle) represent a minimum flow constraint of 2. In (a) A_1 can be directed outward if and only if A_2 and A_3 are both directed inward. In (b) O_1 can be directed outward if and only if any of the other two is directed inward.

of two from G , that is, $c_H(v) := c(v)$. Let $(v, v') \in E$ and let $u \in U$ such that $u \in H(v, v')$. Then $c_H(u) := w(v, v')$. Additionally, suppose that (u, u') represents an edge on the path $H(v, v')$, then we let $w_H(u, u') := w(v, v')$.

Lemma 5. Configuration-to-edge, full-to-full, and edge-to-edge, are PSPACE-complete, even for the grid-embedded constraint graph H that consists of only AND, OR, and connector vertices.

Proof: Every orientation of H can be transformed into an orientation for H , and vice versa. Using this fact the hardness of these three problems immediately follow from Theorem 4. ■

IV. FROM NCL TO MULTI-ROBOT MOTION PLANNING

In this section we present the reduction from the three NCL problems, which were described in the previous section, to our four unlabeled multi-robot motion planning problems; we will call them unlabeled problems for short. Specifically, we consider the case where the input consists of a grid-embedded constraint graph H , as described in Section III. Given such a graph H we construct an unlabeled scenario which corresponds to the graph and consists of unit-square robots and polygonal obstacles. We use a grid layout, as depicted in Figure 3, where the cells of this grid are of dimension 5×5 and the walls separating the cells are of thickness $1/2$. Each such cell functions as a placeholder for a gadget which represents and emulates a specific vertex of H . The gadgets are placed according to the positions of their counterpart vertices in H . Note that between every two adjacent cells there is a doorway of width 1 so that an interaction between adjacent gadgets can take place. We may rotate the gadgets depicted below by 90, 180 or 270 degrees so that gadgets that correspond to two vertices of H that share an edge will share a passage. When two vertices of H share an edge, the corresponding gadgets will share a robot. A similar scheme was employed in [4], although they used different gadgets as they were interested in showing the hardness of slightly different motion planning problems.

A. AND, OR, and connector gadgets

For each vertex of H we create a gadget that emulates the functionality of this vertex in the NCL machine. For the vertices of U we create a *connector* gadget, while for

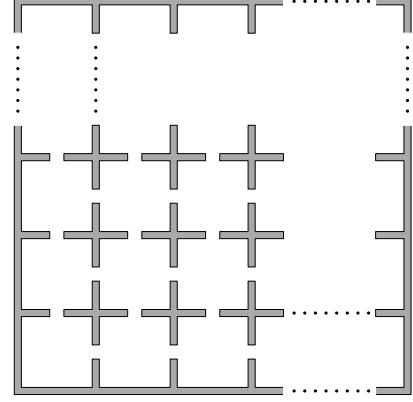


Fig. 3: Placeholders for the gadgets.

the vertices of V we create AND and OR gadgets. All the gadgets fit into 5×5 squares (see Figures 4,5,6) and have either two or three exits through which they connect to other gadgets. Every gadget accommodates several robots and contains polygonal obstacles; the robots are illustrated in purple or green and the obstacles are illustrated in gray. The white regions represent portions of the free workspace. All the robots are placed such that they neither overlap with the obstacles nor with each other. The AND gadgets also have a point obstacle, illustrated in red (its purpose will be explained below). We mention that robots are allowed to touch, but not penetrate, the obstacles. For an illustration of a connection between two gadgets, see Figure 7.

Every gadget accommodates several unit-square robots which fall into two categories: those that never leave the gadget and those that may penetrate the gadget or leave it to a neighbor gadget. The former are called *vertex* robots (drawn in purple), while the latter are *edge* robots (drawn in green). Edge robots are located at the exits of the gadgets, one robot per exit. As the name suggests, edge robots correspond to the edges of H incident to the vertex. The direction of the edge corresponds to the position of the robot, with respect to the gadget. Specifically, an edge that is directed *inward* corresponds to an edge robot that is located at the exit but does not penetrate the 5×5 square of the gadget (for example, robot A_1 in Figure 5a); an *outward* directed edge corresponds to an edge robot located at the exit such that exactly half of it is located inside the 5×5 square of the gadget (see for example robots A_2, A_3 in Figure 5a). We will refer to these

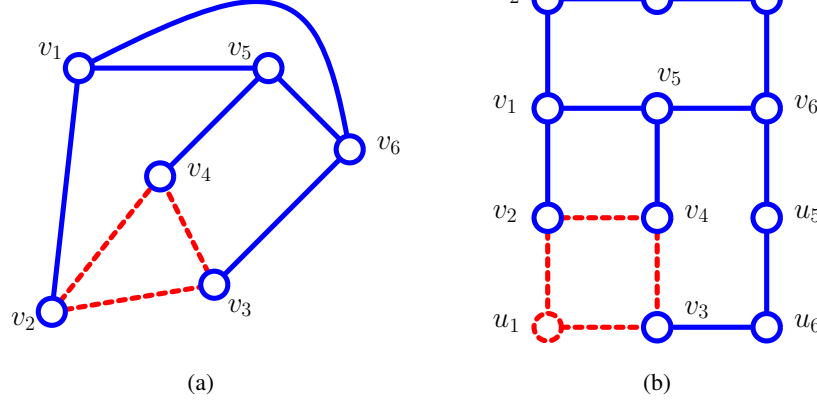


Fig. 2: A grid embedding of a simple planar graph with AND and OR vertices. In (a) we have the original graph $G = (V, E)$ while in (b) we have a grid embedding of it. Recall that blue (solid) edges represent edges with weights of 2, while red (dashed) represent weights of 1. Similarly, blue circles represent vertices with min-flow constraints of 2, while red represent vertices with min-flow constraints of 1. Note that the embedding preserves the type of the vertices from G , e.g., v_4 is an AND vertex in both cases. In order to make the embedding possible the *connector* vertices $U = \{u_1, \dots, u_6\}$ were added to H . Note that the minimum-flow constraint of u_1 is 1. The edges of G are represented as paths in H . For instance, $(v_1, v_6) \in E$ is represented by the path $H(v_1, v_6) = \{v_1, u_2, u_3, u_4, v_6\}$ in H .

two positions of the edge robot as *inside* a gadget, and *outside* a gadget, respectively. We note that when an edge robot of one gadget is located outside, it is also inside the adjacent gadget, and vice versa. The inverse relation between the position of the edge robots and the orientation of the edges stems from the fact that we wish to avoid situations where too few edges are directed into a vertex (and thus the minimum-flow constraint is not satisfied), and situations where too many edge robots are inside a gadget (and thus a collision occurs). For example, in the OR gadget in Figure 6 it is not possible for all the three edge robots to be simultaneously inside the gadget, and this ensures that the corresponding OR vertex in the constraint graph will receive a sufficient amount of in-flow.

For simplicity, we only consider configurations of the robots where the center of each robot is located at the vertices of a $1/2 \times 1/2$ grid. We refer to such configurations as *terminal*. For instance, all the robots in Figures 5,4,6 are placed at terminal configurations. Additionally, the actual terminal configurations are illustrated in Figure 4b. We also allow a robot to move between two adjacent terminal configuration. The following three lemmas illustrate the relation between the gadgets and the vertices of H . Their proofs are straightforward and therefore omitted.

Lemma 6. *Connector gadgets correspond to connector vertices in H , i.e., one of the two edge robot can be inside, if and only if the other edge robot is outside.*

Proof: See Figure 4a and Figure 4b. ■

Lemma 7. *AND gadgets correspond to AND vertices in H , with A_2, A_3, A'_2, A'_3 corresponding to the 1-weight edges, and A_1, A'_1 corresponding to 2-weight edges, e.g., A_1 can move inside the gadget if and only if A_2, A_3 are both outside.*

Proof: See Figure 5a and Figure 5b. ■

Lemma 8. *The OR gadgets correspond to OR vertices in H , i.e., one of the edge robots can move inside if and only if at least one of the other edge robots is outside.*

Proof: See Figure 6. ■

B. Unlabeled motion planning with gadgets

We finalize the details of our reduction and prove its correctness. We first show that the structure of gadgets is very restrictive and allows a limited set of movements for the robots, so that robots cannot wander between different gadgets.

Lemma 9. *Each edge robot can be in at most two distinct terminal configurations.*

Proof: We show that for every possible connection of two gadgets the edge robot can be in at most two terminal configurations. We only consider here the combination of the second AND gadget (Figure 5b) and the OR gadget (Figure 6). The other combinations can be treated analogously. Specifically, we consider the case where the connection is made through the edge robot $A'_3 = O_2$ (colored in orange), as illustrated in Figure 7. Notice that robot $A'_3 = O_2$ is stuck between the robots D and E : D is blocked to the left by the red point obstacle (Figure 7a); E is blocked to the right by an obstacle (Figure 7b). From this observation we conclude that every edge robot can either be inside or outside, and not in any other terminal configuration. ■

Lemma 10. *Each vertex robot can be in at most two distinct terminal configurations, save robot O^* in the OR gadget, which can be in at most three terminal configurations.*

Proof: First, note that every edge robot can move either horizontally or vertically. Let us consider for example robot $A'_3 = O_2$ in Figure 7, which can only move horizontally. Since

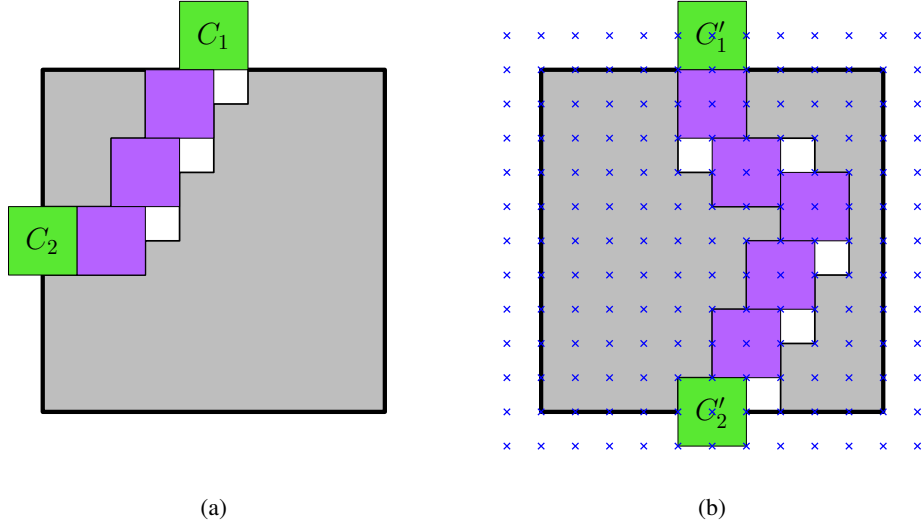


Fig. 4: Connector gadgets. The blue crosses in (b) represent terminal configurations, which are defined in Section IV-A.

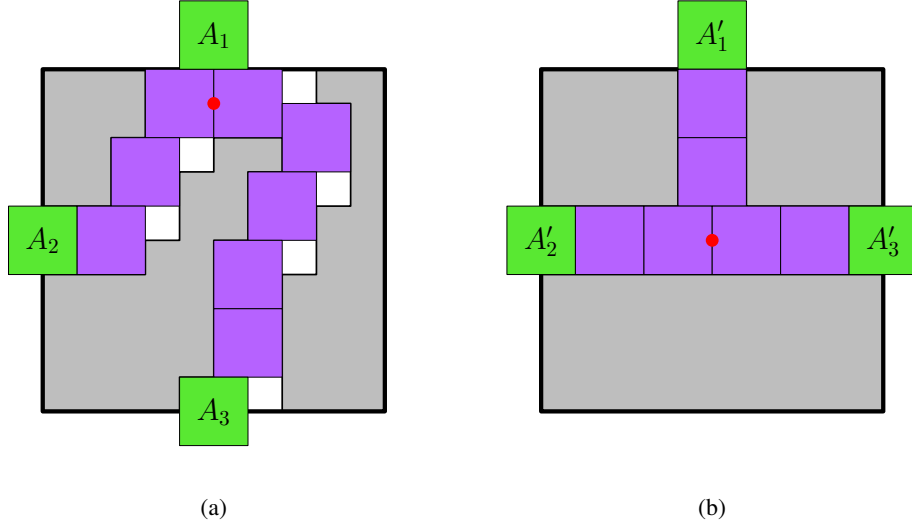


Fig. 5: AND gadgets.

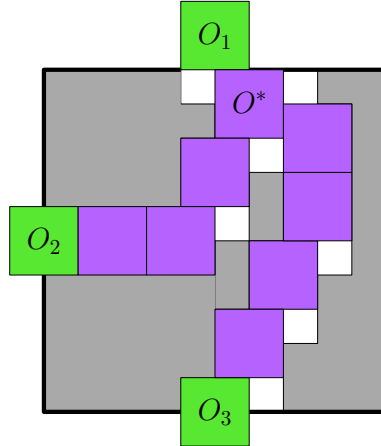


Fig. 6: OR gadget.

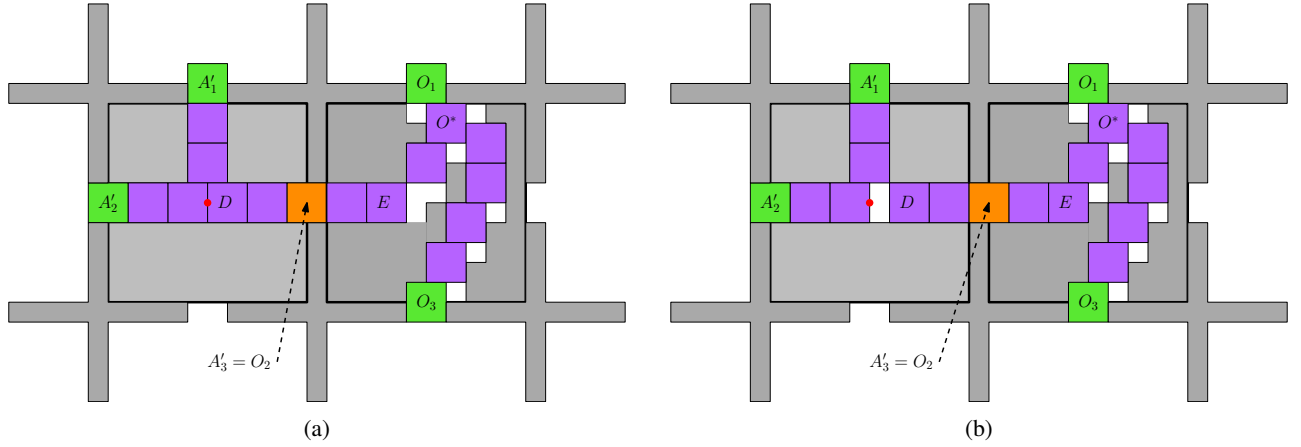


Fig. 7: Illustration for Lemma 9 and a depiction of two gadgets that are connected through an edge robot. In both figures the same AND gadget and OR gadgets, which share an edge robot $A'_3 = O_2$, are illustrated. These two gadgets simulate an AND vertex and an OR vertex which share an edge. For each edge robot A'_i in the AND gadget, denote by a'_i the corresponding edge in the AND vertex. Similarly, denote by o_i the edge corresponding to the edge robot O_i in the vertex. Note that a'_3 and o_2 represent the same edge. In (a) the gadget represents a machine configuration in which a'_1 is directed inward, a'_2 outward, a'_3 outward with respect to the AND vertex. As a consequence, o_2 is directed inward with respect to the OR vertex. Additionally, o_1 is directed inward, while o_3 outward. In (b) a similar machine configuration is depicted, with the exception that a'_3 is now directed inward while o_2 outward.

this robot can only move between two terminal configurations (Lemma 9), the two vertex robot that are directly to its right, cannot move further left than where they appear in Figure 7a. Additionally, robot E is bounded from the right by an obstacle which does not allow it to move further to the right than it appears in Figure 7b. A similar reasoning can be applied to all the other vertex robots. As to robot O^* , consider its position in Figure 7a. It cannot go up as there is an obstacle there. If O_3 leaves the gadget it can move a half step right, and if O_2 leaves the gadget it can move a half step down. Notice however, that it cannot move simultaneously down and right since it will collide with the robot that is immediately to its right. ■

The following lemma implies that our motion planning scenario is so tight that each valid multi-configuration can be interpreted in a single way as an assignment of terminal configurations for the robots.

Lemma 11. *Given a specific terminal configuration, there is at most one robot that can be in it.*

We mention that a similar Lemma can be proven for the gadgets used by Hearn and Demaine [4] in their hardness proof of the labeled variant of the *multi-to-multi* problem. Since their proof uses two different types of robots, their result can also be interpreted as a PSPACE-hardness proof for the 2-color multi-robot motion planning problem [17], which consists of two groups where the robots within each group are identical and interchangeable.

We return to the PSPACE-hardness proof of the four unlabeled problems discussed in our paper.

Theorem 12. *The problems multi-to-multi, multi-to-single, multi-to-single-restricted, and single-to-single, for unit-square robots translating amidst polygonal obstacles in the plane are*

all PSPACE-hard.

Proof: We describe reductions from the three NCL-model decision problems to our four unlabeled problem. Specifically we show the following reductions:

- The *full-to-full* problem, which is concerned with checking whether one NCL machine orientation can be transformed to another, is reduced to the *multi-to-multi* unlabeled problem, which is concerned with deciding whether a collection of robots can be moved between two multi-configurations.
- The *full-to-edge* problem, which is concerned with the existence of an orientation that can be reached from a given orientation where a direction of a specific edge is flipped, is reduced to the unlabeled *multi-to-single* and *multi-to-single-restricted* problems that are concerned with moving an arbitrary or a specific robot (respectively) to a given destination configuration, when a starting multi-configuration is specified for all the robots.
- The *edge-to-edge* problem, which is concerned with the existence of two orientations, where one can be transformed to another, such that the two orientations have an opposite direction for a given edge, is reduced to the unlabeled *single-to-single* problem which is concerned with the existence of two multi-configurations that are equivalent, where each of the two multi-configurations contains a specific configuration that is given as input (where the configuration is not assigned to a specific robot).

We use the same reduction for all three cases. Only the analysis slightly differs. Given a grid-embedded constraint graph H we generate a scenario for the unlabeled problem as we described above, by placing gadgets corresponding to vertices and connecting gadget according to the connections in H .

We first note that given an orientation for H it can be transformed into a valid multi-configuration consisting only of terminal configurations, where the directions of edges in H induce configurations for the respective edge robots, and these in turn induce configurations for the vertex robots. In the other direction, given a valid multi-configuration for the robots that consists of only terminal configurations a valid orientation for H can be easily generated, by considering only the positions of edge robots. Note that by Lemma 9 every edge robot can be in at most two terminal configurations that represent two opposite directions of the same edge of H .

We first prove the hardness of the *multi-to-multi* problem by a reduction from the *full-to-full* problem. Let o_S, o_T be two orientations of H , and denote by S, T the two free multi-configurations induced by them. It is clear that if $o_S \equiv o_T$ then also $S \equiv T$. Now, suppose that $S \equiv T$. Notice that in order to show that this implies that $o_S \equiv o_T$ we need to prove the existence of a solution $\Pi(S, T)$ where no two edge robots move at a given time. More accurately, we need to show that there exists a solution in which every edge robot is in transit between two terminal configurations, only when all the other edge robots are stationary in terminal configurations. We consider for example the AND gadget (Figure 5a) and show that each simultaneous movement of edge robots, where several edge robots are located simultaneously at non-terminal configuration, can be carried out in a sequential manner as well. We treat the various combinations of robots moving simultaneously in and out of the gadget. If both A_2 and A_3 move inside then A_1 must be out, and therefore the former two robots do not depend on each other in order to make their move. Now suppose that A_2 and A_3 simultaneously move outside. This means that each of the two gadgets, to which A_2 and A_3 are entering, already moved other edge robots, and vertex robots, so that the entrance of A_2, A_3 will be possible. Thus the fact that A_2 can leave the gadget does not depend on the fact that A_3 leaves the gadget, and vice versa. Therefore, we can move A_2, A_3 in a sequential manner. Therefore, a solution $\Pi(S, T)$ as required always exists and in the case that $S \equiv T$ it also follows that $o_S \equiv o_T$.

We now proceed to prove the hardness of the *multi-to-single* problem by a reduction from the *full-to-edge* problem. Recall that *full-to-edge* consists of an orientation o_S and edge $e \in E_H$. The question is whether there exists another orientation o_T such that $o_S \equiv o_T$ and $o_S(e) \neq o_T(e)$, i.e., the direction of e in the two configurations is reversed. Denote by S the multi-configuration induced by o_S , and by $s \in S$ the terminal configuration that corresponds to the edge e in the direction $o_S(e)$. We now ask whether there exists a multi-configuration T such that $S \equiv T$ such that there exists $t \in T$ that corresponds to e in the opposite direction. Now that we have defined the components of our *multi-to-single* problem it is clear that if there exists an orientation o_T as required, then its induced multi-configuration satisfies the conditions of *multi-to-single*. The opposite direction follows similarly to the *multi-to-multi* proof above.

The difference between *multi-to-single* and *multi-to-single-restricted* is that in the latter we require that a specific robot will move to a specific target configuration. Note that

our reduction for *multi-to-single* holds here as well, since a selection of a specific edge of H induces the selection of a specific robot that has to move between two configurations (see Lemma 11). The hardness of the *single-to-single* problem by a reduction from the *edge-to-edge* problem can be proved in a manner similar to the previous three cases. ■

We mention that using Lemma 11, our hardness proof can also be applied to prove the hardness of the labeled variant of *multi-to-multi*. In this case, each robot r_i is assigned with specific start and target configurations s_i and t_i , respectively, and the goal is to move each r_i from s_i to t_i while avoiding collisions with robots and obstacles. Specifically, we have the following result.

Theorem 13. *Labeled multi-robot motion planning for unit-square robots moving amidst polygonal obstacles is PSPACE-hard.*

Proof: We follow the same reasoning as for the *full-to-full* to *multi-to-multi* reduction (Theorem 12) and add that due to Lemma 11 given a placement of the robots induced by the starting multi-configuration S , the targets multi-configuration can be viewed as an assignment of a specific target for every robot. Specifically, given an initial assignment s_i for robot r_i , there at most one configuration in T to which it can move (see Lemma 11). ■

V. CONCLUDING REMARKS

In this paper we studied the problem of motion planning of multiple unlabeled unit-square robots in an environment cluttered with polygonal obstacles. We proved that four variants of this problem are PSPACE-hard. While our result in itself is negative, we hope that it will motivate research of other variants of the unlabeled problem which may turn out to be polynomially solvable.

REFERENCES

- [1] Aviv Adler, Mark de Berg, Dan Halperin, and Kiril Solovey. Efficient multi-robot motion planning for unlabeled discs in simple polygons. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2014.
- [2] Kevin Buchin and Maike Buchin. Rolling block mazes are pspace-complete. *JIP*, 20(3):719–722, 2012.
- [3] Kevin Buchin and Dirk H. P. Gerrits. Dynamic point labeling is strongly pspace-complete. In *ISAAC*, pages 262–272, 2013.
- [4] Robert A. Hearn and Erik D. Demaine. PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theoretical Computer Science*, 343(1-2):72–96, 2005.
- [5] Robert A. Hearn and Erik D. Demaine. *Games, puzzles and computation*. A K Peters, 2009.
- [6] Shai Hirsch and Dan Halperin. Hybrid motion planning: Coordinating two discs moving among polygonal obstacles in the plane. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, pages 239–255. Springer, 2002.
- [7] Markus Holzer and Sebastian Jakobi. On the complexity of rolling block and Alice mazes. In *FUN*, pages 210–222, 2012.
- [8] John E. Hopcroft, Jacob T. Schwartz, and Micha Sharir. On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the “Warehouseman’s problem”. *International Journal of Robotics Research*, 3(4):76–88, 1984.
- [9] Takehiro Ito, Marcin Kaminski, and Erik D. Demaine. Reconfiguration of list edge-colorings in a graph. *Discrete Applied Mathematics*, 160(15):2199 – 2207, 2012.
- [10] Stephen Kloder and Seth Hutchinson. Path planning for permutation-invariant multi-robot formations. In *ICRA*, pages 1797–1802, 2005.

- [11] Athanasios Krontiris, Rahul Shome, Andrew Dobson, Andrew Kimmel, Issac Yochelson, and Kostas E. Bekris. Similar part rearrangement with pebble graphs. *CoRR*, abs/1404.6573, 2014.
- [12] Yanpei Liu, Aurora Morgana, and Bruno Simeone. A linear algorithm for 2-bend embeddings of planar graphs in the two-dimensional grid. *Discrete Applied Mathematics*, 81(13):69 – 91, 1998.
- [13] Oren Salzman, Michael Hemmer, and Dan Halperin. On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, pages 313–329, 2012.
- [14] Gildardo Sanchez and Jean-Claude Latombe. Using a PRM planner to compare centralized and decoupled planning for multi-robot systems. In *International Conference on Robotics and Automation (ICRA)*, 2002.
- [15] Jacob T. Schwartz and Micha Sharir. On the piano movers problem: III. Coordinating the motion of several independent bodies. *International Journal of Robotics Research*, 2(3):46–75, 1983.
- [16] Micha Sharir and Shmuel Sifrony. Coordinated motion planning for two independent robots. *Annals of Mathematics and Artificial Intelligence*, 3(1):107–130, 1991.
- [17] Kiril Solovey and Dan Halperin. k -Color multi-robot motion planning. *International Journal of Robotic Research*, 33(1):82–97, 2014.
- [18] Kiril Solovey, Oren Salzman, and Dan Halperin. Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2014.
- [19] Kiril Solovey, Jingjin Yu, Or Zamir, and Dan Halperin. Motion planning for unlabeled discs with optimality guarantees. In *Robotics: Science and Systems (RSS)*, 2015. To appear.
- [20] Paul G. Spirakis and Chee-Keng Yap. Strong NP-hardness of moving many discs. *Information Processing Letters*, 19(1):55–59, 1984.
- [21] Matthew Turpin, Nathan Michael, and Vijay Kumar. Concurrent assignment and planning of trajectories for large teams of interchangeable robots. In *International Conference on Robotics and Automation (ICRA)*, pages 842–848, 2013.
- [22] Petr Švestka and Mark H. Overmars. Coordinated path planning for multiple robots. *Robotics and Autonomous Systems*, 23:125–152, 1998.
- [23] Glenn Wagner and Howie Choset. M*: A complete multirobot path planning algorithm with performance bounds. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 3260–3267, 2011.
- [24] Chee-Keng Yap. Coordinating the motion of several discs. Technical report, Courant Institute of Mathematical Sciences, New York, 1984.